[Document Name] Specification

[Title of the Invention] Image Processor

[Scope of Claims for Patent]

[Claim 1] An image processor comprising:

a plurality of processors processing input image data in parallel with each other and outputting processed said image data; and

an address memory storing address information of said image data processed by each of said plurality of processors.

[Claim 2] The image processor according to claim 1, further comprising:

an image memory storing said image data output from said plurality of processors, and

read means reading said image data from said image memory on the basis of said address information stored in said address memory.

[Claim 3] The image processor according to claim 1, further comprising an image memory storing said image data output from said plurality of processors along the sequence of addresses on the basis of said address information stored in said address memory.

[Claim 4] The image processor according to any of claims 1 to 3, further comprising:

input means inputting image data subjected to processing in synchronization with a first external device, and

output means outputting said image data processed in said plurality of processors and said address information stored in said address memory in synchronization with a second external device.

[Detailed Description of the Invention]

[Technical Field to Which the Invention Belongs]

The present invention relates to an image processor, and more particularly, it relates to an image processor comprising a plurality of processors processing input

image data in parallel with each other and outputting the processed image data.

[Prior Art]

Figs. 23(A) is a block diagram showing the structure of an image processor provided on a conventional digital copying machine. Referring to Fig. 23(A), the image processor is formed by a CCD 901, an A-D conversion part 903, a shading correction and LOG conversion part 905, a variable power part 907, an MTF correction part 909, a γ correction part 911 and a binarization processing part 913.

The A-D conversion part 903 converts image data subjected to photoelectric conversion by the CCD 901 to digital data. Thereafter the image data is sequentially subjected to image processing and then output to a printer or the like. The blocks for the image processing perform synchronous pipeline processing to operate with the same clock.

Fig. 23(B) is a block diagram showing the structure of an image processor performing image processing with an MPU 917. This image processor employs the MPU 917 in place of the image processing blocks 905 to 913 shown in Fig. 23(A). Such a device is capable of rewriting an image processing program at any time and hence has a high degree of freedom, although the same is inferior to the device employing synchronous pipeline processing in consideration of the operating speed under the present circumstances. The processing time for each pixel varies with the load of processing due to asynchronous processing with the MPU 917. Therefore, this device requires an input memory 915 and an output memory 919 for the MPU 917, for the purpose of synchronization with an input unit and an output unit.

Japanese Patent Laying-Open No. 3-177961 discloses a multiprocessor control unit. This processes image data for one frame in block units with a plurality of MPUs of a parallel structure.

Fig. 24 is a block diagram showing the structure of such a multiprocessor control unit. The multiprocessor control unit is formed by a data flow control part 952, a feedback frame memory 951, a status register 953, MPU1 to MPU4, a feedback bus,

an output image bus and an input image bus.

The data flow control part 952 monitors values of the status register 953 storing processing states of the MPU1 to MPU4 and busy conditions of the buses. The data flow control part 952 instructs each MPU to operate. In other words, the data flow control part 952 allocates a new block of image data to an MPU terminating processing. A free MPU sequentially processes image data sequentially input in a serial manner, so that the image data are output at random regardless of the input sequence.

When requiring a result of processing of a preceding frame, each MPU captures feedback data of a required area from the feedback frame memory 951 through the feedback bus.

In the device shown in Fig. 24 processing and outputting subsequent input data from the MPU terminating processing through the data flow control part 952, the MPU has no idling standby time. In this device outputting image data regardless of the sequence of input data, however, the output data must be rearranged.

In order to eliminate the necessity for such rearrangement of the output data, the data output timings of the MPUs may be adjusted. Such a method is now described.

Fig. 25 shows a state of image data of one frame divided into 4 by 6 (= 24) blocks. In this case, the MPU1 to MPU4 process the blocks of columns L1 to L4 respectively. Referring to Fig. 26, data of blocks D00, D01, D02 and D03 are input in the multiprocessor control unit at a time T1, so that the MPU1 to MPU4 process the data of the blocks D00 to D03 respectively. At a time t2 when the MPU (MPU3 in this case) having the lowest processing speed terminates the processing, processed data Q00 to Q03 are sequentially output. At the same time, data of new blocks D10 to D13 are input at the time t2 and processed by the MPU1 to MPU4 respectively.

Thus, the data can be output along the sequence of the blocks due to such processing, to require no rearrangement.

[Problems to be Solved by the Invention]

In the aforementioned method shown in Fig. 26, however, the MPUs have idling

standby times although no rearrangement of the output data is necessary, and hence the image processing cannot be performed at a high speed.

Accordingly, an object of the present invention is to provide an image processor capable of performing image processing at a high speed and readily rearranging output data.

[Means for Solving the Problems]

In order to attain the aforementioned object, an image processor according to an aspect of the present invention comprises a plurality of processors processing input image data in parallel with each other and outputting the processed image data and an address memory storing address information of the image data processed by each of the plurality of processors.

Preferably, the image processor further comprises an image memory storing the image data output from the plurality of processors and read means reading the image data from the image memory on the basis of the address information stored in the address memory.

Preferably, the image processor further comprises an image memory storing the image data output from the plurality of processors along the sequence of addresses on the basis of the address information stored in the address memory.

More preferably, the image processor further comprises input means inputting image data subjected to processing in synchronization with a first external device and output means outputting the image data processed in the plurality of processors and the address information stored in the address memory in synchronization with a second external device.

According to the present invention, the address information of the image data processed by each of the plurality of processor is so stored that an image processor capable of performing image processing at a high speed and readily rearranging output data can be provided.

[Embodiments]

[First Embodiment]

Fig. 1 is a block diagram showing the structure of an image processor according to a first embodiment of the present invention. Referring to Fig. 1, the image processor is formed by a data flow control part 101, a status register 103, a feedback frame memory 105, MPU1 to MPU4, a feedback bus, an output image bus, an address bus outputting addresses of output image data, an input image bus and an address bus inputting addresses of input image data.

The image data of a single image subjected to processing is divided into a plurality of image data, which in turn are processed by the MPU1 to MPU4 in parallel with each other. Memories (address memories) M1 to M4 storing address information (address data) of the image data processed by the respective ones of the MPU1 to MPU4 are provided for the MPU1 to MPU4 respectively.

When the image data processed by the MPU1 to MPU4 are output, the address data are also output. Thus, processing for restoring a single image from the output image data is simplified.

The output memory 919 may be provided on the rear stage of the device for writing corresponding processed image data in addresses of the output memory 919 corresponding to the address data stored in the memories M1 to M4, as shown in Fig. 23(B). Thus, a single processed image can be obtained.

Fig. 2 illustrates a specific example of the status register 103. The status register stores what the states of the respective MPU1 to MPU4 are. Each MPU has five states, i.e., "standby", "input", "processing", "end of processing" and "output" states.

The MPU performs no processing in the "standby" state, and receives image data from the input image bus in the "input" state. The MPU processes the input image data in the "processing" state, and terminates processing the image data in the "end of processing" state. The MPU outputs the image data through the output image bus in the "output" state.

The status register indicates the current state of each MPU with "1".

Fig. 3 is a flowchart showing processing of the data flow control part 101. Referring to Fig. 3, the data flow control part 101 confirms the contents (see Fig. 2) of the status register 103 at a step S101. The data flow control part 101 determines whether or not any MPU is in the "standby" state at a step S103. If the determination is of YES, the data flow control part 101 instructs the MPU to receive and process image data at a step S105. The data flow control part 101 determines whether or not processing of all image data is terminated at a step S107, and ends this routine if the determination is of YES.

If the determination at the step S103 is of NO, the process advances to the step S107. If the determination at the step S107 is of NO, the process returns to the step S101.

Fig. 4 is a flowchart showing processing of one of the MPU1 to MPU4. Referring to Fig. 4, the MPU enters the "standby" state at a step S201. This state is recorded in the status register 103.

A determination is made at a step S203 as to whether or not an instruction is received from the data flow control part 101, and the processing from the step S201 is repeated until the determination is of YES.. If the determination at the step S203 is of YES, the state of the MPU in the status register 103 is changed at a step S205. More specifically, "0" is written in the column of "standby" and "1" is written in the column of "input".

At a step S207, the MPU inputs image data subjected to processing by the MPU through the input image bus. At a step S209, the MPU stores the address of the image data processed by the MPU in the corresponding one of the memories M1 to M4.

At a step S211, "0" is set in the column of "input" in the status register 103 while "1" is set in the column of "processing". The input image is processed at a step S213. If the processing is terminated, "0" is set in the column of "processing" and "1" is set in the column of "end of processing" in the status register at a step S215.

The MPU enters a standby state at a step S217, to wait for an output instruction

from the data flow control part 101 at a step S219. When receiving the output instruction, "0" is set in the column of "end of processing" and "1" is set in the column of "output" in the status register at a step S221. At a step S223, the image data processed by the MPU is output through the output image bus, along with the address data stored by the corresponding one of the memories M1 to M4 through the address bus. At a step S225, "0" is set in the column of "output" and "1" is set in the column of "standby" in the status register. Thereafter the process returns to the step S201.

Fig. 5 is a timing chart showing the relation between the input data, the processing of the MPU1 to MPU4, the output data and the address data. Referring to Fig. 5, the image data D00 to D03 are input in the MPU1 to MPU4 respectively at a time T1 for starting image processing. The MPU2 terminating the processing first outputs the image data Q01 as an output result of the image data Q01. At this time, address data A01 for the image data Q01 is also output. Thereafter the image data D10 subjected to subsequent processing is input in the MPU2, which in turn starts processing the image data D10. Then, an MPU subsequently completing processing outputs data so that new image data is input in this MPU. Such processing is so repeated that the standby times of the MPU1 to MPU4 can be reduced as compared with the prior art shown in Fig. 26, and the image processing can be performed at a higher speed. While the sequence of the output data is different from that of the input data, the output data, which are output along with the address data, can be efficiently rearranged through the address data.

While each MPU captures feedback data of a required area from the feedback memory 105 through the feedback bus, when requiring the result of processing of a preceding frame, only a single feedback memory 105 is provided on the device to be shared by the MPU1 to MPU4 according to this embodiment, and hence the burden of processing is reduced as compared with the case of providing such feedback memories for the MPU1 to MPU4 respectively.

[Second Embodiment]

Fig. 6 is a block diagram showing the structure of an image processor according to a second embodiment of the present invention. This image processor 201 is connected between a scanner (image reader) and a printer, for processing data while communicating with an MPU 202 of the scanner and an MPU 203 of the printer.

The image processor 201 is formed by a data flow control part 205, a status register 207, a selector 209, address FIFO memories 211a to 211d, image data FIFO memories 213a to 213d, MPU1 to MPU4, address FIFO memories 215a to 215d, image data FIFO memories 217a to 217d, buffer memories 219 to 225 and a selector 227.

The selector 209 inputs image data subjected to processing in synchronization with the scanner serving as an external device. The selector 227 outputs the processed image data and address information in synchronization with the printer serving as an external device.

The status register 207 stores the states of the MPUs, as described with reference to Fig. 2.

The image processor 201 according to this embodiment is provided with the FIFO memories 213a to 213d and 217a to 217d for the image data and the FIFO memories 211a to 211d and 215a to 215d for the address data in correspondence to synchronous input and output.

The data flow control part 205 having a counter checks the quantities of input and output data while making communication with the scanner MPU 202 and the printer MPU 203. The data flow control part 205 interrupts data input when the FIFO memories 211a to 211d and 213a to 213d are full, while interrupting data output when the FIFO memories 215a to 215d and 217a to 217d are vacant.

Fig. 7 is a block diagram showing the structure around the selector 227 in detail. Referring to Fig. 7, the selector 227 includes a MIN value detection circuit 229 and transfer gates 231a to 231d.

Fig. 8 is a flowchart showing operations of the selector 227. Referring to Fig. 8, the selector 227 activates a read signal RE of each of the FIFO memories 215a to

215d and 217a to 217d for one pixel in response to an output start instruction from the data flow control part 205 at a step S301. The selector 227 stores read data in each of the buffer memories 219 to 225. The MIN value detection circuit 229 detects the minimum one from address data stored in the buffer memories 219 to 225 and outputs a common read signal to the FIFO memory, the buffer memory and the transfer gate corresponding to the minimum address data. Thus, image data corresponding to the minimum address data is selected as output data, and data from the FIFO memories are newly stored in the buffer memories 219 to 225. This operation is performed in synchronization with an output clock. Thus, image data stored in the FIFO memories at random are sequentially sorted by an output unit and output along the sequence of the addresses.

The image processor 201 according to this embodiment can be synchronized with an input unit (e.g., the scanner) and the output unit (e.g., the printer) through the FIFO memories. Further, processing in the MPU1 to MPU4 can be performed independently of (asynchronously with) the input unit and the output unit. Thus, image processing can be performed regardless of data transfer clocks of the input and output units.

[Third Embodiment]

Fig. 9 is a block diagram showing the structure of an image processor according to a third embodiment of the present invention. Referring to Fig. 9, the image processor is formed by an input buffer memory 301 temporarily storing input image data, a CPU 303 controlling the overall device, MPU1 to MPU3 dividing input image data into a plurality of image data and processing the same in parallel with each other, line memories L1 and L2 storing output image data, a memory controller 305 controlling the line memories L1 and L2 and a count register 307 counting the quantities of image data written in the line memories L1 and L2. The memory controller 305 includes registers for storing addresses of the image data processed by the MPU1 to MPU3 respectively and the employed line memory L1 or L2.

Fig. 10 illustrates the structure of the registers included in the memory controller 305. Referring to Fig. 10, each register is formed by (N + 1) bits. The registers are provided in correspondence to the number (three in this embodiment) of the MPUs.

Each register records lower N bits of the image data processed by each MPU. As shown in Fig. 11, the lower N bits of the image data specify the position (line memory address) of a pixel in image data D for one line recorded in each line memory. As shown in Fig. 10, the registers record the line memory L1 or L2 storing the image data by 1-bit data together.

When one of the line memories L1 and L2 has a capacity of 4 kilobytes, for example, the address width thereof is 12 bits and hence each line memory is formed by a 13-bit register.

Fig. 12 is a flowchart showing processing of the CPU 303. Referring to Fig. 12, the CPU 303 confirms a status flag provided on each MPU thereby searching for an MPU performing no processing. The processing from the step S401 is repeated until a determination is made that there is an MPU performing no processing at the step S403.

If the determination at the step S403 is of YES, the CPU 303 outputs image data to the MPU performing no processing and instructs this MPU to process the image data at a step S405.

The CPU 303 writes the lower N bits of the address of the image data processed by the MPU and one of the line memories L1 and L2 for writing the image data in the register of the memory controller 305 at a step S407.

The CPU 303 determines whether or not all image data are completely processed at a step S409, to return to the step S401 if the determination is of NO or complete this routine if the determination is of YES.

Fig. 13 is a flowchart showing processing of each of the MPU1 to MPU3. Referring to Fig. 13, the MPU enters a standby state (state performing no processing) and waits for an instruction from the CPU 303 at a step S501. If an instruction is received from the CPU 303, the MPU reads data subjected to processing from the input

buffer memory 301 at a step S503. The MPU performs image processing at a step S505, and determines whether or not the processing is terminated at a step S507. If the processing is terminated, the MPU outputs the result of the processing to the memory controller 305 at a step S509. Thereafter the MPU repeats the processing from the step S501.

Fig. 14 is a flowchart showing processing of the memory controller 305. Referring to Fig. 14, the memory controller 305 initializes the count register 307 at a step S601. At a step S603, the memory controller 305 receives processed data from any of the MPU1 to MPU3 through the CPU 303.

At a step S605, the memory controller 305 writes the processed data received from the MPU in the corresponding address of the line memory L1 or L2 on the basis of the line memory address stored in the register (see Fig. 10) of the memory controller 305.

At a step S607, the memory controller 305 stores the number of data written in the line memory in the count register 307. More specifically, the memory controller 305 increments the value of the count register 307 by one. At a step S609, the memory controller 305 determines whether or not the value of the count register 307 reaches a prescribed value (N bits). This is processing for determining whether or not the line memory is full. If the determination is of YES, the memory controller 305 instructs the line memory L1 or L2 to output data by the prescribed value (N bits). Thereafter the process returns to the step S601.

If the determination at the step S609 is of NO, the process returns to the step S603.

Fig. 15 illustrates the processing of the MPUs and the write procedure of each line memory in the image processor according to this embodiment. Referring to Fig. 15, the input buffer memory 301 receives image data D0 to D4095 and address data 0 to 4095. The MPU1 to MPU3 sequentially process the data stored in the input buffer memory 301. Each MPU terminating the processing receives new data from the input

buffer memory 301 and processes the same. The data, which are written in the line memory from the processed one, are not necessarily written along the sequence of addresses. However, the output data are written in positions indicated by address data, thereby allowing rearrangement.

The count register 307 performs counting every time data is written in the line memory. When the value of the count register 307 reaches a prescribed value (4095 in this embodiment) to indicate that the line memory is filled with data, the image data stored in the line memory are output.

Fig. 16 illustrates a write operation in the line memory. Referring to Fig. 16, it is assumed that the line memory records n image data 1 to n. In other words, the line memory is so controlled as to output recorded image data when the count register counts the value n.

As shown at (A), it is assumed that image data of an address (n – 1) of the line memory is unprocessed while the remaining image data are completely processed and written in the line memory. At this time, the value of the count register 307 is (n – 1). If any MPU completely processes the image data of the address (n – 1), this image data Dn–1 is written in the address (n – 1) of the line memory as shown at (B). At this time, the value of the count register is incremented by one to reach n. Then, the memory controller 305 outputs an output signal in synchronization with a clock signal CLK. Thus, data are sequentially output from the line memory along the sequence of D1 to Dn, as shown at (C). Thereafter the line memory is initialized to clear all data recorded therein, as shown at (D).

According to this embodiment, the processed data from the MPUs are written in the line memory and hence results asynchronously processed by the MPUs can be efficiently output in association with a synchronous signal. Further, the image data can be output in line units without waiting for completion of processing of image data for one page stored in the input buffer memory, whereby high-speed processing is enabled.

In addition, the memory controller controls the storage state of the line memory,

and hence the burden of the CPU can be reduced.

[Fourth Embodiment]

Fig. 17 is a block diagram showing the structure of an image processor according to a fourth embodiment of the present invention.

Referring to Fig. 17, the image processor is formed by an input buffer memory 401 storing input image data, MPU1 to MPU3 dividing the input image data into a plurality of image data and processing the same in parallel with each other, a data flow control part 409 controlling the MPU1 to MPU3, an output buffer memory 403 temporarily storing output image data, an address memory 405 storing addresses of the output image data, and a memory controller 407 reading the image data from the output buffer memory 403 on the basis of the address data stored in the address memory 405 and making control to output the image data along the sequence of the addresses.

Fig. 18 illustrates operations of the image processor according to this embodiment. The input buffer memory stores image data (a1 to f1) along the sequence of addresses (1 to 7). The MPU1 to MPU3 process the image data in parallel with each other, and hence the sequence of the image data stored in the output buffer memory does not necessarily match with the sequence of the image data stored in the input buffer memory. Therefore, the image processor includes the address memory 405 storing the addresses of the image data stored in the output buffer memory 403, thereby simplifying rearrangement of the image data. In other words, the memory controller 407 reads the image data from the output buffer memory 403 along the sequence of the addresses stored in the address memory 405 and outputs the image data. Thus, rearranged image data are output.

Fig. 19 is a flowchart showing processing of the data flow control part 409. Referring to Fig. 19, the data flow control part 409 confirms free states of the MPUs at a step S701. When a determination is made that there is a free MPU at a step S703, the data flow control part 409 instructs the free MPU to perform processing at a step S705. Thereafter the data flow control part 409 determines whether or not all data are

completely processed, and terminates the processing if the determination is of YES.

If the determination at the step S703 is of NO, the process returns to the step S701. If the determination at the step S707 is of NO, the process also returns to the step S701.

Fig. 20 is a flowchart showing operations of each MPU. Referring to Fig. 20, the MPU enters a standby state at a step S801, to wait for an instruction from the data flow control part. When receiving an instruction from the data flow control part 409, the MPU reads image data subjected to processing from the input buffer memory 401 at a step S803. The MPU processes the image data at a step S805, and waits for termination of the processing at a step S807. If the processing is terminated, the MPU stores the processed image data in the output buffer memory 403 and writes the address of the image data in the corresponding position of the address memory at a step S809. Thereafter the process returns to the step S801.

While the MPUs sequentially write the data in the output buffer memory and the address memory from that terminating processing in this embodiment, writing may be performed on the basis of predetermined priority when a plurality of MPUs simultaneously terminate processing.

Fig. 21 is a flowchart showing operations of the memory controller 407.

Referring to Fig. 21, "counter" indicates a counter counting a data output number. "Address T" indicates addresses of the address memory 405 and the output buffer memory 403. "Data ADR" indicates data stored in the address memory 405. "Data Dout" indicates data stored in the output buffer memory 403.

Referring to Fig. 21, the memory controller 407 initializes the counter at a step S901. This counter corresponds to the data stored in the output buffer memory 403, and counts up along the sequence of 0, 1, 2, ... every time data is output.

At a step S903, the memory controller 407 initializes the address T. Then, the memory controller 407 reads the data ADR of the address T from the address memory.

The memory controller 407 determines whether or not the data ADR and the

value of the counter match with each other at a step S907, and outputs the data Dout of the address ADR of the output buffer memory at a step S909 if the determination is of YES. Then, the memory controller 407 increments the value of the counter by one at a step S911, and determines whether or not all image data are completely output at a step S913.

If the determination at the step S913 is of YES, this routine is terminated. If the determination at the step S913 is of NO, the process returns to the step S903.

If the determination at the step S907 is of NO, the memory controller 407 increments the address T of the address memory by one at a step S915, and the process returns to the step S905.

Fig. 22 illustrates effects of the image processor according to this embodiment.

Referring to Fig. 22, the addresses T of the address memory and the output buffer memory correspond to each other. The address memory sequentially stores addresses (addresses of input data) n−1, n, ... of image data subjected to processing. The output buffer memory stores output data Dn−1, Dn, ... obtained as the result of image processing.

After all data are processed, the address memory is sequentially searched for addresses of the data to be output. For example, the address memory is searched for the data (n − 1). The address for the data (n − 1) in the address memory is (i + k), and hence the output buffer memory outputs data of the address (i + k).

Then, the address memory is searched for data n, in order to output n-th data. The address of the address memory for the data n is i, and hence the output buffer memory outputs data of the address i.

Output results stored at random from the head can be sequentially output by performing the aforementioned operations until all data N are output.

As hereinabove described, the image processor according to this embodiment can efficiently sequentially output data asynchronously processed and stored at random.

While images in page units must be divided into block units in general image

processing, each MPU in each of the first to fourth embodiments of the present invention performs processing in block units (it is assumed that one pixel forms one block in the second to fourth embodiments) of the image data. In the first to fourth embodiments of the present invention, therefore, effective processing suitable for image processing is enabled.

The embodiments disclosed this time must be considered as illustrative in all points and not restrictive. The scope of the present invention is shown not by the above description but by the scope of claim for patent, and it is intended that all modifications within the meaning and range equivalent to the scope of claim for patent are included.

[Brief Description of the Drawings]

[Fig. 1] A block diagram showing the structure of an image processor according to a first embodiment of the present invention.

[Fig. 2] A diagram showing the structure of a status register.

[Fig. 3] A flowchart showing processing of a data flow control part.

[Fig. 4] A flowchart showing processing of a single MPU.

[Fig. 5] A diagram for illustrating effects of the first embodiment.

[Fig. 6] A block diagram showing the structure of an image processor according to a second embodiment of the present invention.

[Fig. 7] A block diagram showing the structure of a selector 227 in Fig. 6.

[Fig. 8] A flowchart showing operations of the selector 227.

[Fig. 9] A block diagram showing the structure of an image processor according to a third embodiment of the present invention.

[Fig. 10] A diagram showing the structure of a memory controller 305.

[Fig. 11] A diagram for illustrating a line memory address.

[Fig. 12] A flowchart showing processing of a CPU.

[Fig. 13] A flowchart showing processing of a single MPU.

[Fig. 14] A flowchart showing processing of the memory controller.

[Fig. 15] A diagram for illustrating effects of the third embodiment.

[Fig. 16] A diagram for illustrating write and output timings of a line memory.

[Fig. 17] A block diagram for illustrating the structure of an image processor according to a fourth embodiment of the present invention.

[Fig. 18] A diagram for illustrating effects of the fourth embodiment.

[Fig. 19] A flowchart showing processing of a data flow control part.

[Fig. 20] A flowchart showing processing of a single MPU.

[Fig. 21] A flowchart showing operations of a memory controller.

[Fig. 22] A diagram for illustrating effects of the fourth embodiment.

[Fig. 23] Block diagrams showing the structures of conventional image processors.

[Fig. 24] A block diagram showing the structure of an image processor having a plurality of MPUs arranged in parallel with each other.

[Fig. 25] A diagram showing specific examples of image data processed by the image processor.

[Fig. 26] A diagram for illustrating problems of the prior art.

[Description of the Reference Characters]

101 data flow control part

103 status register

105 feedback memory

211a to 211d address FIFO memory

213a to 213d image data FIFO memory

215a to 215d address FIFO memory

217a to 217d image data FIFO memory

205 data flow control part

207 status register

301 input buffer memory

303 CPU

305 memory controller

307 count register

MPU1 to MPU4 MPU

M1 to M4 address memory